Assignment 1

Name: Chirag Sarda

Roll No.: 20110047

The code are pushed on the github

Coding Question

I have used IPv4 and TCP protocol for the connection \

```
server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server_socket.bind(addr_tuple)
# AF_INET --- address family of IPv4
# SOCK_STREAM --- connection oriented tcp protocol
```

User can run the server_code.py first, it will give the IP address and port number. After we can run client_code.py, and then specify the IP address and port number of the server.

After that client code will ask the user to specify the crypto mode. One can give 1,2 and 3 corresponding to modes given in the question. Crypto layer is attacked at the zeroth index of the sending string, and it is not encrypted and decrypted. Server and Client can get to know which crypto mode is used for by reading the zeroth index.

```
PS E:\5thSem\CS433\Assignment\Assignment1\CLIENT> python .\client_code.py
[RUNNING] Client started running
IP address of the host - 192.168.56.1
Port no of the host - 5050
[CREATED] clinet socket successfully created
[REQUESTING] client requesting for connection
Which cypto mode you want to use 2
User command -
```

```
PS E:\5thSem\CS433\Assignment\Assignment1\SERVER> python .\server_code.py
[RUNNING] Server running at IP = 192.168.56.1  and port number = 5050
[LISTENING] server is listening...
[CONNECTION SUCCESSFUL] Server accepts the connection of client at ('192.168.56.1', 56120)
```

After that it will ask for the user to give command:

1. CWD command: Client is sending the encrypted message (according to the crypto mode chosen earlier)

Client:

```
User command - CWD
[SENDING] CWD commnad to Server
Sending packet to the server  2EYF
[RESPONSE] server response: E:\5thSem\CS433\Assignment\Assignment1\SERVER
User command -
```

Server:

```
The packet received at the server is  2EYF
[RECEIVED] CWD commnad at Server
[SENT] Server sent response to client
```

2. LS command:

Client:

```
User command - LS
[SENDING] LS commnad to Server
Sending packet to the server  2NU
[RESPONSE] server response: a.exe
Home
Library
server_code.py
User
User command -
```

Server:

```
The packet received at the server is  2NU
[RECEIVED] LS commnad at Server
[SENT] Server sent response to client
```

3. CD Command

Client:

```
User command - CD User
[SENDING] CD commnad to Server
Sending packet to the server  2EF Wugt
[RESPONSE] server response: [OK]
User command -
```

Server:

```
The packet received at the server is  2EF Wugt
[RECEIVED] CD commnad at Server
[SENT] Server sent response to client
```

Checking whether our CD command is successful or not

```
User command - CD User
[SENDING] CD commnad to Server
Sending packet to the server  2EF Wugt
[RESPONSE] server response: [OK]
User command - CWD
[SENDING] CWD commnad to Server
Sending packet to the server  2EYF
[RESPONSE] server response: E:\5thSem\CS433\Assignment\Assignment1\SERVER\User
User command -
```

4. UPD Command

Client:

```
User command - UPD gt.txt
[SENDING] UPD commnad to Server
[NOK] File not found at the client
User command - UPD yt.txt
[SENDING] UPD commnad to Server
Sending packet to the server  2WRF av.vzv vjku ku vjg av vgzv hkng^&&*7
[RESPONSE] server response: [OK]
User command - LS
[SENDING] LS commnad to Server
Sending packet to the server  2NU
[RESPONSE] server response: ht.txt
yt.txt
User command -
```

Server:

```
The packet received at the server is  2EYF
[RECEIVED] CWD commnad at Server
[SENT] Server sent response to client
The packet received at the server is  2WRF av.vzv vjku ku vjg av vgzv hkng^&&*7
[RECEIVED] UPD commnad at Server
[SENT] Server sent response to client
The packet received at the server is  2NU
[RECEIVED] LS commnad at Server
[SENT] Server sent response to client
```

We can check whether the file is being uploaded or not by checking list of all the file (LS command) if file is not found it will display error ([NOK] file not found at the client)

As we can see that it returns yt.txt in the last line which means our upload was correct

5. DWD command

Client:

```
User command - LS
[SENDING] LS commnad to Server
Sending packet to the server  2NU
[RESPONSE] server response: ht.txt
yt.txt
User command - DWD htt.txt
[SENDING] DWD commnad to Server
Sending packet to the server  2FYF jvv.vzv
File Not Found on the server
User command - DWD ht.txt
[SENDING] DWD commnad to Server
Sending packet to the server  2FYF jv.vzv
ht.txt downloaded successfully
User command -
```

Server:

```
[RECEIVED] LS commnad at Server
[SENT] Server sent response to client
The packet received at the server is  2FYF jvv.vzv
[RECEIVED] DWD commnad at Server
[SENT] Server sent response to client
The packet received at the server is  2FYF jv.vzv
[RECEIVED] DWD commnad at Server
[SENT] Server sent response to client
```

All the downloaded file saves in the Current working directory of client, if the file is not found on the server then it will display file not found on the server

6. exit(): to the close the connection between the server and client

Client:

```
Which cypto mode you want to use 1
User command - LS
[SENDING] LS commnad to Server
Sending packet to the server  1LS
[RESPONSE] server response: a.exe
Home
Library
server_code.py
User
User command - ls
[SENDING] ls commnad to Server
Sending packet to the server  1ls
[RESPONSE] server response: Command Not Found
User command - xds
[SENDING] xds commnad to Server
Sending packet to the server  1xds
[RESPONSE] server response: Command Not Found
User command -
```

Server:

```
The packet received at the server is  1LS
[RECEIVED] LS commnad at Server
[SENT] Server sent response to client
The packet received at the server is  1ls
[RECEIVED] ls commnad at Server
[SENT] Server sent response to client
The packet received at the server is  1xds
[RECEIVED] xds commnad at Server
[SENT] Server sent response to client
```

7. else situation: If some random command is typed: Code will display message (Command not found), but connection will not get close.

Client

```
User command - CWD
[SENDING] CWD commnad to Server
Sending packet to the server  1CWD
[RESPONSE] server response: E:\5thSem\CS433\Assignment\Assignment1\SERVER
User command - fldfld
[SENDING] fldfld commnad to Server
Sending packet to the server  1fldfld
[RESPONSE] server response: Command Not Found
User command -
```

Server:

```
The packet received at the server is  1CWD
[RECEIVED] CWD commnad at Server
[SENT] Server sent response to client
The packet received at the server is  1fldfld
[RECEIVED] fldfld commnad at Server
[SENT] Server sent response to client
```

Sending DWD in all the 3 different modes

1: Plain Text

Client:

```
PS E:\5thSem\CS433\Assignment\Assignment1\CLIENT> python .\client_code.py
[RUNNING] Client started running
IP address of the host - 192.168.56.1
Port no of the host - 5050
[CREATED] clinet socket successfully created
[REQUESTING] client requesting for connection
Which cypto mode you want to use 1
User command - CWD
[SENDING] CWD commnad to Server
Sending packet to the server  1CWD
[RESPONSE] server response: E:\5thSem\CS433\Assignment\Assignment1\SERVER
User command - CD Home
[SENDING] CD commnad to Server
Sending packet to the server  1CD Home
[RESPONSE] server response: [OK]
User command - LS
[SENDING] LS commnad to Server
Sending packet to the server  1LS
[RESPONSE] server response: a.cpp
C1.txt
C2.txt
C3.txt
gt.txt
User command - DWD C1.txt
[SENDING] DWD commnad to Server
Sending packet to the server  1DWD C1.txt
C1.txt downloaded successfully
User command -
```

Server:

```
PS E:\5thSem\CS433\Assignment\Assignment1\SERVER> python .\server_code.py
[RUNNING] Server running at IP = 192.168.56.1  and port number = 5050
[LISTENING] server is listening...
[CONNECTION SUCCESSFUL] Server accepts the connection of client at ('192.168.56.1', 56549)
The packet received at the server is  1CWD
[RECEIVED] CWD commnad at Server
[SENT] Server sent response to client
The packet received at the server is  1CD Home
[RECEIVED] CD commnad at Server
[SENT] Server sent response to client
The packet received at the server is  1LS
[RECEIVED] LS commnad at Server
[SENT] Server sent response to client
The packet received at the server is  1DWD C1.txt
[RECEIVED] DWD commnad at Server
[SENT] Server sent response to client
```

2. Substitution Mode:

Client:

```
PS E:\5thSem\CS433\Assignment\Assignment1\CLIENT> python .\client_code.py
[RUNNING] Client started running
IP address of the host - 192.168.56.1
Port no of the host - 5050
[CREATED] clinet socket successfully created
[REQUESTING] client requesting for connection
Which cypto mode you want to use 2
User command - CWD
[SENDING] CWD commnad to Server
Sending packet to the server  2EYF
[RESPONSE] server response: E:\5thSem\CS433\Assignment\Assignment1\SERVER
User command - CD Home
[SENDING] CD commnad to Server
Sending packet to the server  2EF Jqog
[RESPONSE] server response: [OK]
User command - LS
[SENDING] LS commnad to Server
Sending packet to the server  2NU
[RESPONSE] server response: a.cpp
C1.txt
C2.txt
C3.txt
gt.txt
User command - DWD C2.txt
[SENDING] DWD commnad to Server
Sending packet to the server  2FYF E4.vzv
C2.txt downloaded successfully
User command -
```

Server:

```
PS E:\5thSem\CS433\Assignment\Assignment1\SERVER> python .\server_code.py
[RUNNING] Server running at IP = 192.168.56.1  and port number = 5050
[LISTENING] server is listening...
[CONNECTION SUCCESSFUL] Server accepts the connection of client at ('192.168.56.1', 56559)
The packet received at the server is  2EYF
[RECEIVED] CWD commnad at Server
[SENT] Server sent response to client
The packet received at the server is  2EF Jqog
[RECEIVED] CD commnad at Server
[SENT] Server sent response to client
The packet received at the server is  2NU
[RECEIVED] LS commnad at Server
[SENT] Server sent response to client
The packet received at the server is  2FYF E4.vzv
[RECEIVED] DWD commnad at Server
[SENT] Server sent response to client
```

3 Transpose Mode:

Client:

```
PS E:\5thSem\CS433\Assignment\Assignment1\CLIENT> python .\client_code.py
[RUNNING] Client started running
IP address of the host - 192.168.56.1
Port no of the host - 5050
[CREATED] clinet socket successfully created
[REQUESTING] client requesting for connection
Which cypto mode you want to use 3
User command - CWD
[SENDING] CWD commnad to Server
Sending packet to the server  3DWC
[RESPONSE] server response: E:\5thSem\CS433\Assignment\Assignment1\SERVER
User command - CD Home
[SENDING] CD commnad to Server
Sending packet to the server  3DC emoH
[RESPONSE] server response: [OK]
User command - LS
[SENDING] LS commnad to Server
Sending packet to the server  3SL
[RESPONSE] server response: a.cpp C1.txt C2.txt C3.txt gt.txt
User command - DWD C3.txt
[SENDING] DWD commnad to Server
Sending packet to the server  3DWD txt.3C
C3.txt downloaded successfully
User command - exit()
[SENDING] exit() commnad to Server
Sending packet to the server  3)(tixe
[RESPONSE] server response: exit()
[CLOSED] Clinet connection closed
PS E:\5thSem\CS433\Assignment\Assignment1\CLIENT>
```

Server:

```
PS E:\5thSem\CS433\Assignment\Assignment1\SERVER> python .\server_code.py
[RUNNING] Server running at IP = 192.168.56.1  and port number = 5050
[LISTENING] server is listening...
[CONNECTION SUCCESSFUL] Server accepts the connection of client at ('192.168.56.1', 56578)
The packet received at the server is  3DWC
[RECEIVED] CWD commnad at Server
[SENT] Server sent response to client
The packet received at the server is  3DC emoH
[RECEIVED] CD commnad at Server
[SENT] Server sent response to client
The packet received at the server is  3SL
[RECEIVED] LS commnad at Server
[SENT] Server sent response to client
The packet received at the server is  3DWD txt.3C
[RECEIVED] DWD commnad at Server
[SENT] Server sent response to client
The packet received at the server is  3)(tixe
[RECEIVED] exit() commnad at Server
[SENT] Server sent response to client
[CLOSED] server connection closed
PS E:\5thSem\CS433\Assignment\Assignment1\SERVER>
```

Challenges

1. Once we do the CD command then after that we do CWD it was giving the original directory not the that was changed by the previous CD command. For this have declared a current_working_directory variable before the connection is formed.
2. Transporting the file across the socket (UPD and DWD)
3. Applying the crypto Layer (How to decide when to apply which crypto layer). How server will know which crypto Layer we have to use. For this I just added the mode at the 0^{th} index so that server and client will no which crypto mode we are using

4. Making the connection persistent. For that I used the while loop. Received
5. Sending the file is easy as the data is in string format one can directly apply the crypto modes but when it comes to image and video file there, we have to transport information in bytes (in binary), it becomes very complex to handle the crypto mode. My code does not handle this

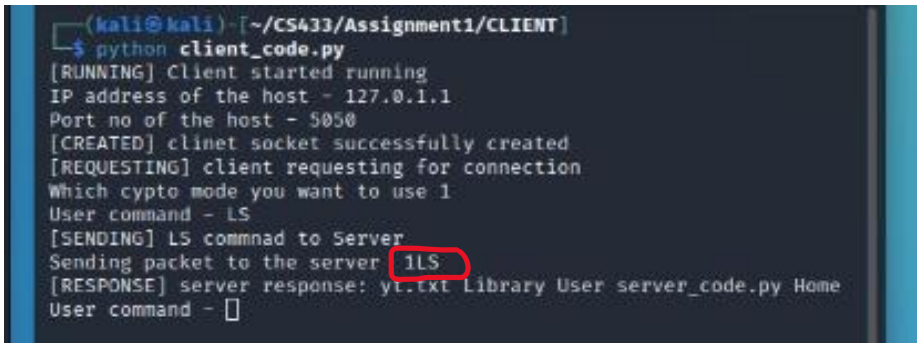
Checking for the correctness of the encryption layer by wireshark:

For this I have runt the code on the kali as I have wireshark on the kali in Virtual Box:

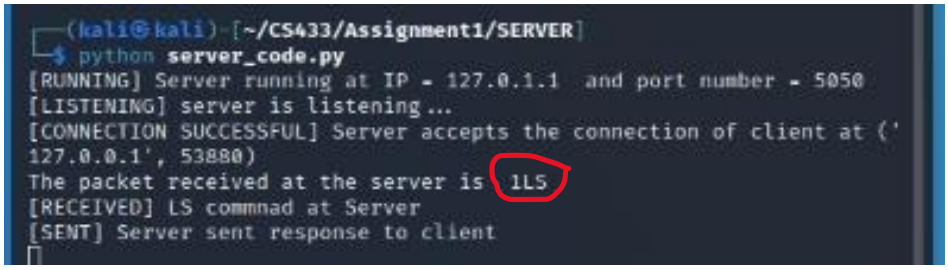
Using the LS command for the analysis:

Crypto Mode 1:

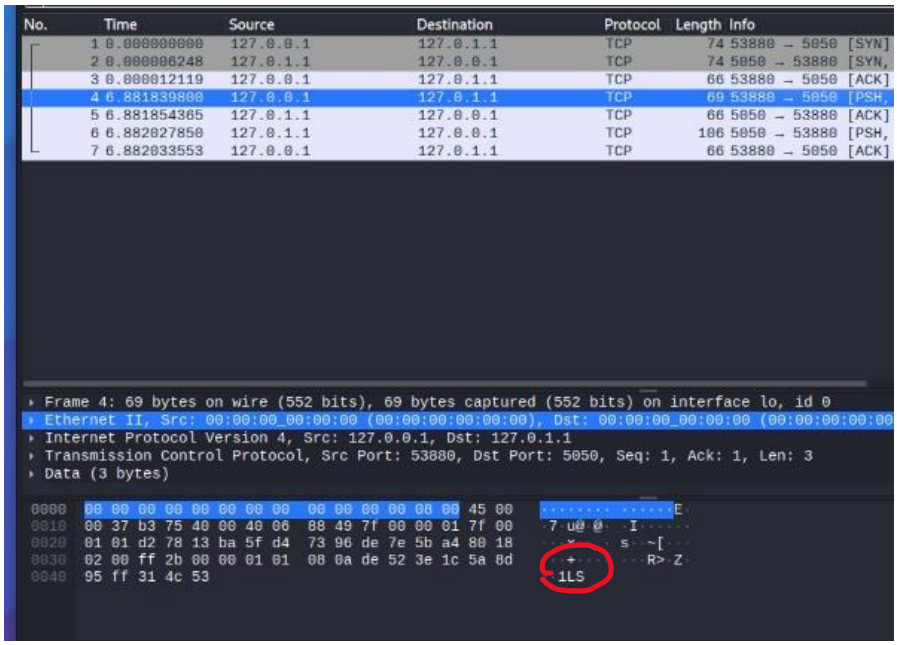
Client:



Server:



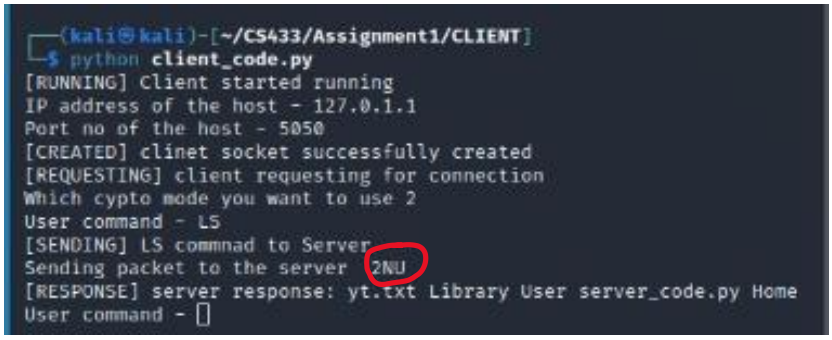
Wireshark:



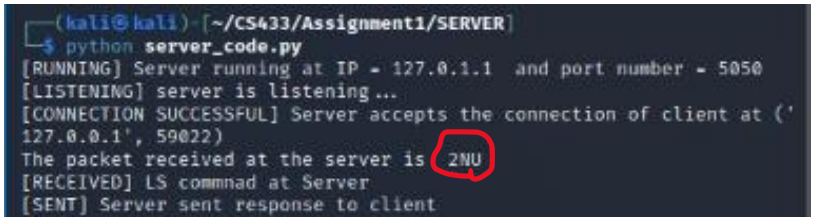
In wireshark and in my code both are showing 1LS, 1 is for crypto mode

Crypto mode = 2

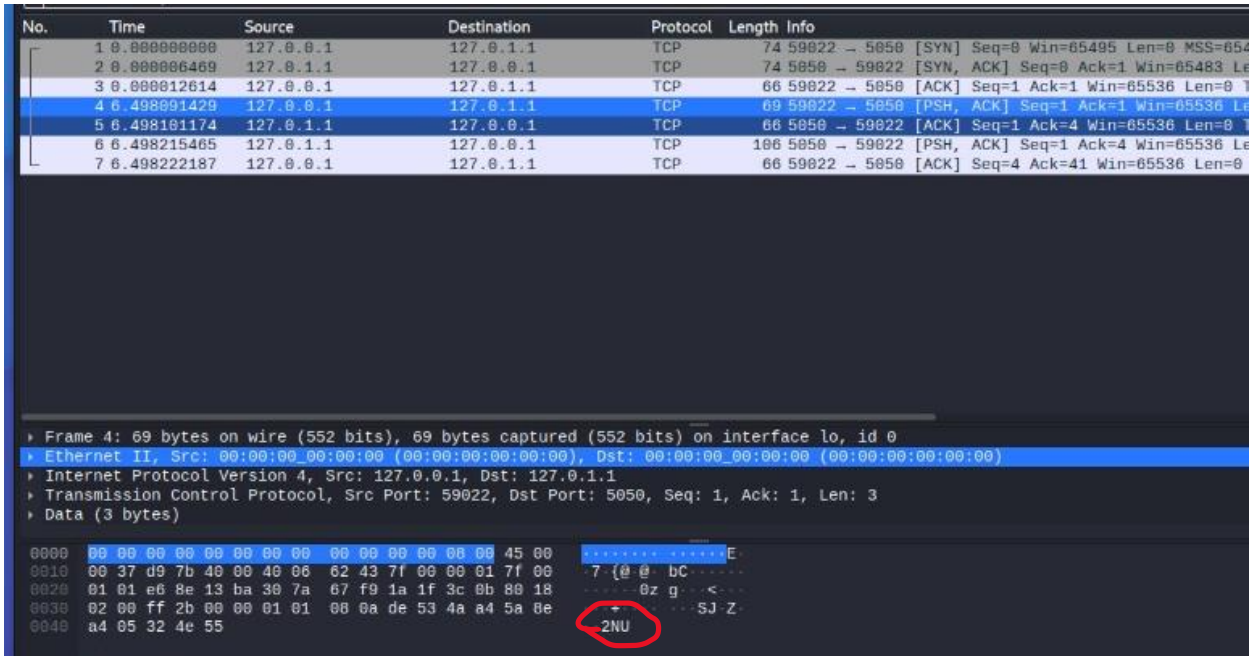
Client:



Server:



Wireshark:



When LS is changed to NU in by Crypto mode of substitution. Wireshark and our code gives the same answer

Crypto Mode = 3

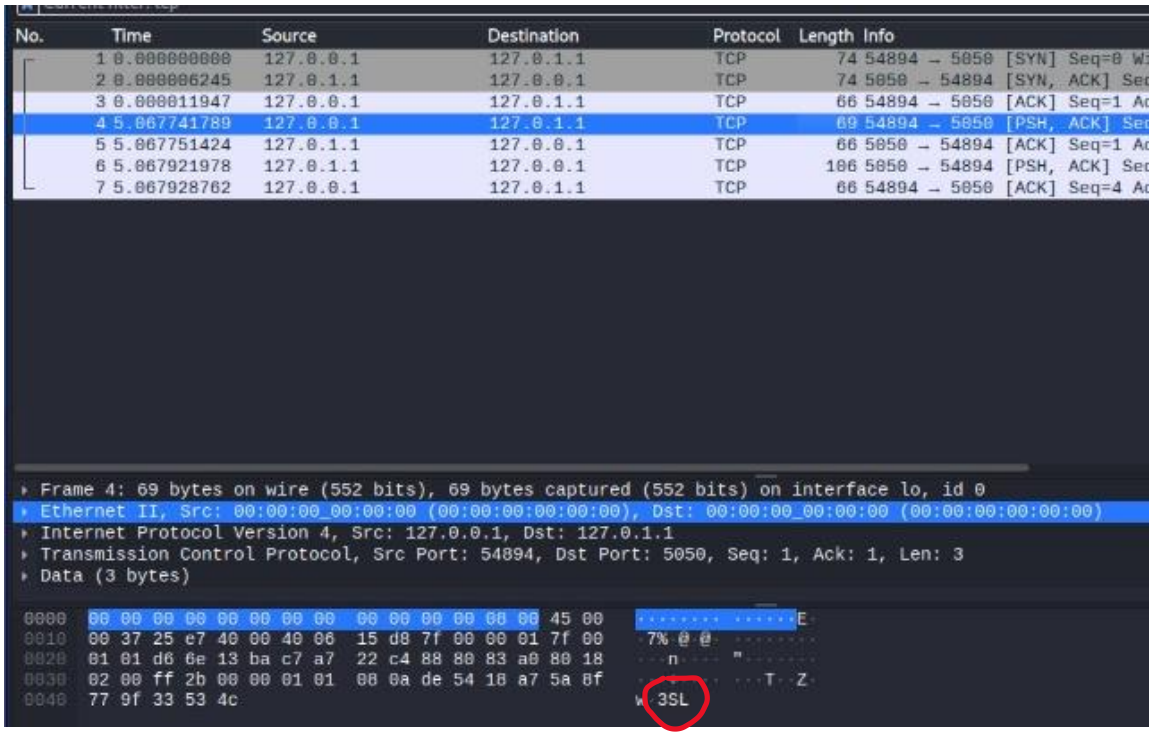
Client:

```
┌──(kali㉿kali)-[~/CS433/Assignment1/CLIENT]
└─$ python client_code.py
[RUNNING] Client started running
IP address of the host - 127.0.1.1
Port no of the host - 5050
[CREATED] clinet socket successfully created
[REQUESTING] client requesting for connection
Which cypto mode you want to use 3
User command - LS
[SENDING] LS commnad to Server
Sending packet to the server 3SL
[RESPONSE] server response: yt.txt Library User server_code.py Home
User command -
```

Server:

```
┌──(kali㉿kali)-[~/CS433/Assignment1/SERVER]
└─$ python server_code.py
[RUNNING] Server running at IP - 127.0.1.1  and port number - 5050
[LISTENING] server is listening...
[CONNECTION SUCCESSFUL] Server accepts the connection of client at ('
127.0.0.1', 54894)
The packet received at the server is  3SL
[RECEIVED] LS commnad at Server
[SENT] Server sent response to client
```

Wireshark:



When LS is changed to SL in by Crypto mode of substitution. Wireshark and our code gives the same answer

For other commands we can similarly do this.

apsara

Date:

Numerical Question

Name : Chirag Sardg
Roll no : 2011047

Question

→ Message Size = 100 KB
Meta data = 100 B / packet

Given → propagation delay = 0 s
processing delay = 0 s

(a) 1 packet
→ No need to worry for other packet as there is only one packet

packet Size = Message Size + 100 B
= 100 kB + 100 B
= (100 000 + 100) B
= (100100) B
= 100.1 kB

Here T is Transmission delay

Total Time to Reach B = $T_{Link1} + T_{Link2} + T_{Link3}$

$$T_{delay} = \frac{\text{Packet Size}}{\text{Bandwidth}}$$

$$T_{Link1} = \frac{100.1\ kB}{400\ MB/s} = \frac{1.001 \times 10^5 \times 8}{4 \times 10^8}\ s$$

$$= 2.002\ ms$$

$$T_{Link2} = \frac{100.1\ kB}{100\ MB/s} = 4(2.002\ ms) = 8.008\ ms$$

$$T_{Link3} = \frac{100.1\ kB}{200\ Mb/s} = 2(2.002) = 4.004\ ms$$

Page No.:

apsara

Date:

Total Time = 2.002 + 8.008 + 4.004

= 14.014 ms

(b) → 10 packets

→ Total packet Size = 100 B + $\frac{10 \text{ kB}}{\text{packet}}$

= 100 B + 10,000 B

= 10.1 kB

→ Calculating the Transmission delay for individual Link.

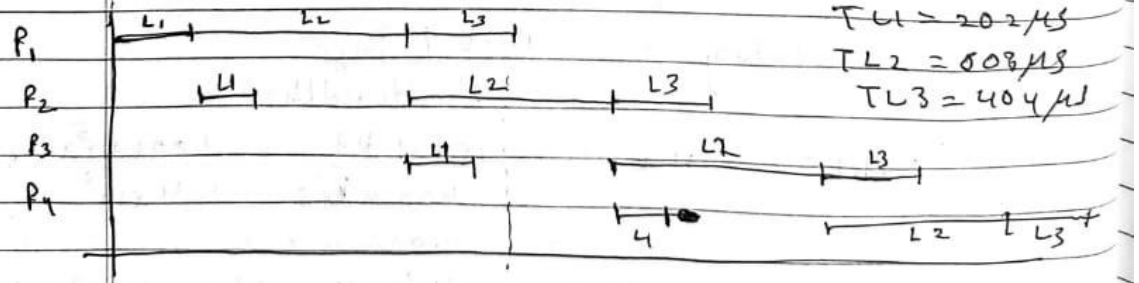
for Link 1 = $\frac{10.1 \text{ kB}}{400 \text{ Mb/s}}$ = $\frac{10.1 \times 10^3 \times 8}{400 \times 10^6}$ = 0.202 ms

= 202 μs

for Link 2 = $\frac{10.1 \text{ kB}}{100 \text{ Mb/s}}$ = 4(202 μs) = 808 μs

for Link 3 = $\frac{10.1 \text{ kB}}{200 \text{ Mb/s}}$ = 2(202 μs) = 404 μs

For total Time lets look at way individual packets arrive



TL1 = 202 μs

TL2 = 808 μs

TL3 = 404 μs

if 4 packets then $TL_1 + 4 \times TL_2 + TL_3$

Page No:

apsara

Date:

→ Although the diagram looks confusing as it suggest the Propagation delay but it is not

→ it is showing Gantt Chart

→ P1 leaves A at 202 μs, leaves B at 808 μs leaves C at 404 μs (It needs not to worry about others.

Packet 2

→ P2 can enter A after P1 has left but it can't leave A until P1 has left R1

→ This causes the P1 to wait which increase our time

→ For 10 packets

Total Time = 202 μs + 10 (806 μs) + 404 μs

= (8080 + 202 + 404) μs

= 8686 μs

= 8.686 ms

(c) 50 packets

Total packet size = 100 B + $\frac{100}{50}$ KB = 2.1 KB

Transmission delay

Link 1 = $\frac{2.1 \text{ KB}}{400 \text{ Mbs}} = \frac{2.1 \times 10^3 \times 8}{400 \times 10^6}$ = 42 μs

Link 2 = 4 (Link 1) = 168 μs

Link 3 = 2 (Link 1) = 84 μs → any packet count = 50

Total Time = (42 + 50 × 168 + 84) μs = ms

= 8.526 ms

Page No.:

(a) packet Count = 100
packetSize = 100 B + $\frac{100\ kB}{100}$ = 1.1 kB

Transmission delay for
Link 1 = $\frac{1.1\ kB}{400\ Mb/s}$ = $\frac{1.1 \times 10^3 \times 8}{400 \times 10^6}$ = 22 μs
Link 2 = 4 (Link 1) = 88 μs
Link 3 = 2 (Link) = 44 μs

Total Time 22 + 100(88) + 44
22 + 8800 + 44
8866 μs
8.866 ms

Clearly out of 8.526 ms is the lowest delivery time.

(3) → R1 ——— R2
10 Km
Bandwidth = 100 Gbps
$v = \frac{2}{3}c = \frac{2}{3} \times 3 \times 10^8 = 2 \times 10^8$ m/s

(a) → Propagation delay = $\frac{\text{Length}}{\text{velocity}}$ = $\frac{50 \times 10^3}{2 \times 10^8}$ = 50 μs
Delay in Medium while transferring

(b) R1 ——— R2

Maximum bit in the medium (Link)
= Propagation time × Bandwidth
= $100 \times 10^9 \times 50 \times 10^{-6}$ = 5×10^6 b
= 5 Mb

(c) ~~Distance~~
Bit Width = ~~Speed~~ Speed / Bandwidth = $\frac{2 \times 10^8}{10^{11}}$ =
= 2 mm

(d) → RTT (Round Trip Time) = 40 ms
Web page size → 1 kB
Object Size → 100 kB
10 Object Load

(e) HTTP (1.0) Non Persistent Connection
→ $\sum_{i=1}^{n}$ (2RTT + file transmission time)
n = ~~total Object loads~~
2RTT → 1 initial TCP/UDP connection (since http so mainly TCP)
2 → http request/ Response

(f) ~~Non Persistent~~
Individual
|| (2RTT + file transmission time)
↳ 1 for web page
↳ 10 for its object

= 22RTT + t (Assuming total transmission time be t it will include the time for web page & 10 object
= 220ms + t
for web page
t = t_1 + 10 t_2
for object

(b) Persistent Connection (HTTP 1.1)

Total Time = RTT + $\sum_{i=1}^{M}$ (RTT + file transfer)

RTT → Initial Connection

RTT → http connection

= (1 + 11) RTT + t → All file transfer time

= 12 RTT + t

= 120 ms + t

t = t_1 + 10 t_2

t_1 → for Web page

t_2 → for object loaded by web page

(c) → HTTP (2.0) (persistent + pipelined) → data frames of 1 KB

Total Time = RTT + RTT + RTT + file transfs time

RTT → Initial TCP

RTT → http for web page

RTT → http for object

30 ms + t

→ t_1 would depends on size of web page similarly t_2 would depends on size of Object

If assumped $t_1 \propto$ Size

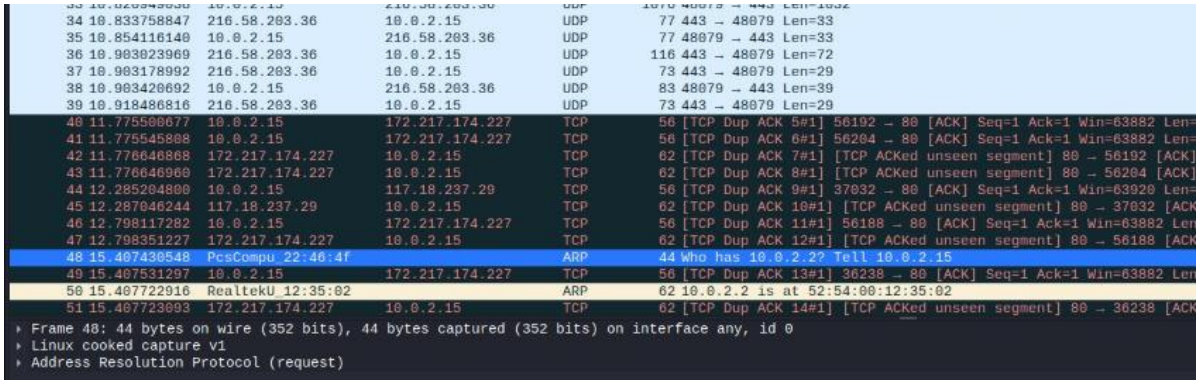
For Non persistent = Total Time = 220 + (1 + 1000) t'

persistent = 120 + (1 + 1000) t'

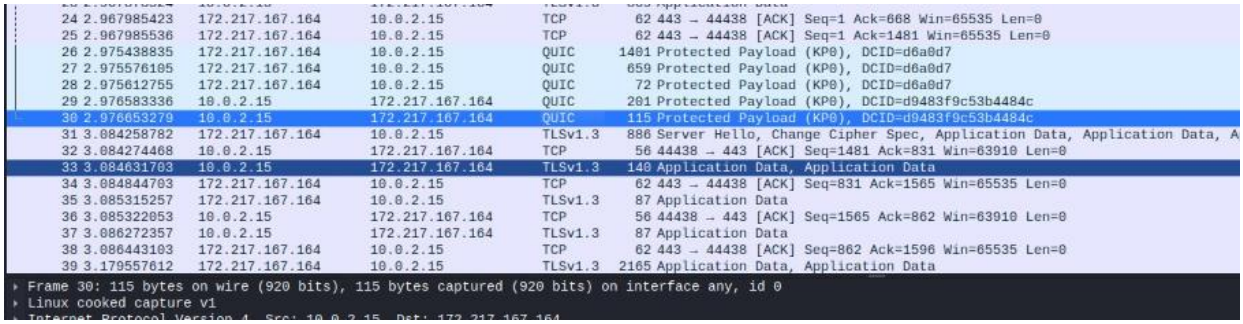
pipeline + persistent = 30 ms + 1001 t'

New

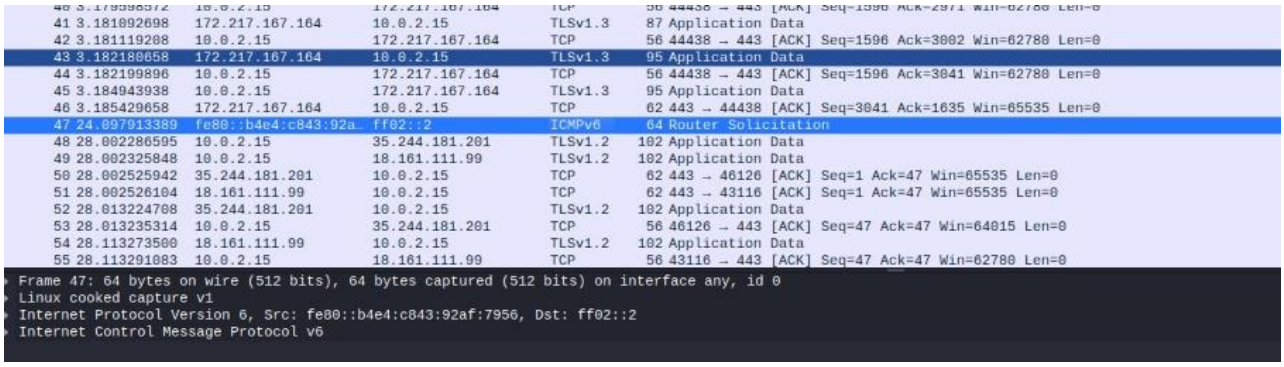
1. ARP Address Resolution Protocol RFC 826 is used to convert the IP address into physical address



2.**QUIC:** RCF 8999 to 9002, new reliable and secure transport layer protocol, made to replace the TCP

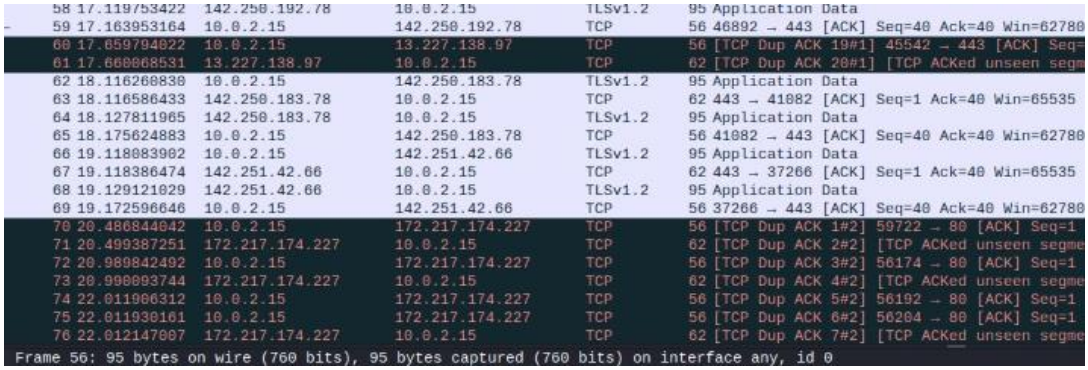


3. **ICMP**: Internet Control Message Protocol (RFC 792), used for host to host datagram services in the interconnected network

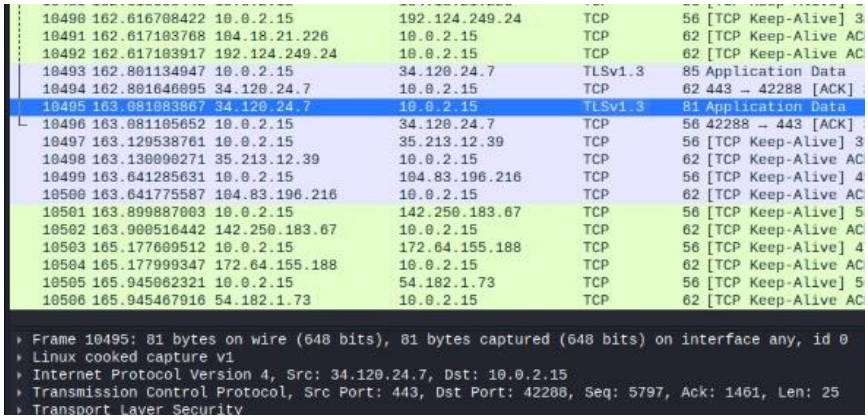


4.TLSv1.v2: RFC 5426: Transport layer security protocol. Provides com

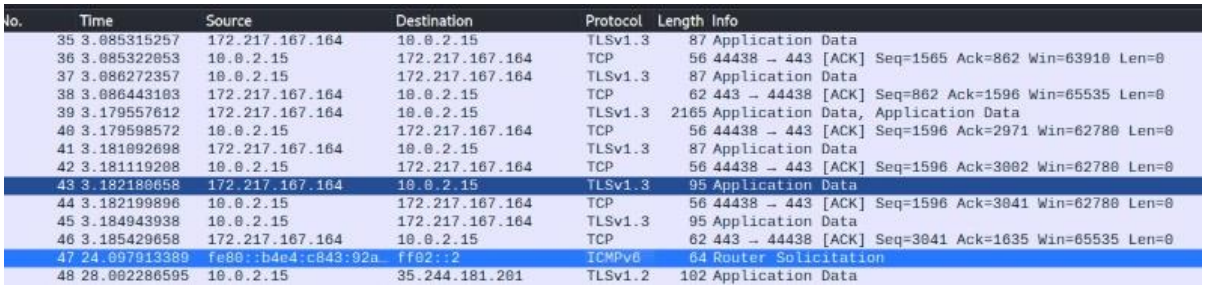
munaication security on the internet



TLC v1.v3: it is designed to prevent eavesdropping, tampering, and message forgery



Estimating the RTT:



We will choose 41 and 42th packet

We can clearly see that our local host (10.0.2.15) started a TCP connection to the server (172.217.167.164) and then it send the message as TLSc1.3

RTT = time when 43rd packet stated moving – time when 41st packet started moving

= 3.182199896 – 3.181892698

= 0.3072 ms

This tells us that it is very fast